

MODÜL 8

TURTLE (KAPLUMBAĞA) MODÜLÜ İLE GRAFİK ARAYÜZE GİRİŞ



Şekil 8.1: Bölümle ilgili örnek uygulamalara karekod' dan ulaşabilirsiniz

Bu bölüm, bilgisayar ekranlarından aşına olduğunuz pencereler (formlar) ile tanışacağınız ve kullanmaya başlayacağınız kısımdır. Grafik arayüz (pencere) kavramı, temelde yazılımın komut satırından çıkıp görsel ara birimler aracılığıyla ifade edilmesidir. Kaplumbağa grafikler grafik arayüz üzerinde çizim araçları kullanarak çalışılmasına olanak tanır. Kaplumbağa grafikler Python gibi metin tabanlı dillerde kod yazmaya yeni başlayanlar için en eğlenceli modüllerden biridir. Kaplumbağa grafikler ile daha önceki bölümlerde öğrendiğiniz kavramları kullanarak bol miktarda pratik yapabilirsiniz. Kaplumbağa grafiklerin tarihine bakıldığında matematik eğitimi için logo programla dilinde kullanıldığı görülmektedir. John Dewey'in öğrencisi olan Seymour Papert, öğrencilerine matematik öğretmek için kaplumbağa robotunu geliştirmiştir.

MODÜL 8

Turtle (**kaplumbağa**) standart python modüllerinden biridir. Bu nedenle turtle modülünü çalışmamızda kullanabilmek için modülü içeri aktarmamız (dâhil etmemiz) gerekmektedir. Bunun için ilk satıra **import turtle** yazmanız yeterli olacaktır. Python turtle modülünü çağırırken **turtle.py** adında bir dosya çağırıldığından yapılacak örneklere turtle.py isminin verilmemesi yararlı olacaktır. turtle modülü projeye dâhil edildikten sonra bir turtle nesnesi oluşturup buna isim verilir. Bunu çizim yapmak için bir kalem almak gibi düşünebilirsiniz. **Bu bölümdeki uygulamalar idle kullanılarak yapılmıştır.**

8.1. Turtle (kaplumbağa) ile Çalışma

kalem=turtle.Turtle() yazarak kalem isminde bir Turtle (kaplumbağa) nesnesi oluşturmuş oluruz. Kalem artık Turtle nesnesinin sahip olduğu tüm özelliklere sahip olacaktır. Böylece Turtle nesnesinin sahip olduğu fonksiyonlar kullanılarak çizim işlemleri yapılabilir. Nesnenin fonksiyonlarını yazarken nesne adından sonra nokta koyup fonksiyon adı yazılır. Örneğin, **kalem.forward()** kalem nesnesinin ileri gitme fonksiyonudur. forward fonksiyonu gidilecek mesafeyi girdi olarak alır. Örneğin **kalem.forward (50)** yazılarak elli birim ilerlenir. Çizim işlemi bitince **turtle.done()** fonksiyonu ile program tamamlanır. Böylece turtle modülü kullanarak kod yazmaya hazır hale gelinir. Örnek 1'deki dört satırlık kodu yazıp çalıştırdığınızda sonuç Şekil 8.2'de görüldüğü gibi olur.

Örnek 1

100 birimlik ileri yönde çizgi çizme

```
import turtle
kalem=turtle.Turtle()
kalem.forward(100)
turtle.done()
```



Şekil 8.2: Örnek 1 kod çıktısı

8.2. Temel Hareket İşlemleri

Turtle modülünde dört temel hareket bulunmaktadır. Bunlar, ileri: **forward (mesafe)**, geri: **backward (mesafe)**, sağ: **right (açı)**, sol: **left (açı)** komutlarıdır. Bu komutlar kullanılarak yapılan örnekleri inceleyelim.

Örnek**2**

right ve forward fonksiyonlarının birlikte kullanımı

```
import turtle
kalem=turtle.Turtle()
kalem.forward(100)
kalem.right(90)
kalem.forward(100)
turtle.done()
```

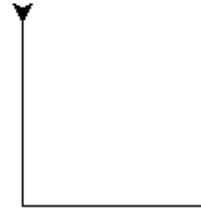


Şekil 8.3: Örnek 2 kod çıktısı

Örnek**3**

right ve backward fonksiyonlarının birlikte kullanımı

```
import turtle
kalem=turtle.Turtle()
kalem.backward(100)
kalem.right(90)
kalem.backward(100)
turtle.done()
```



Şekil 8.4: Örnek 2 kod çıktısı

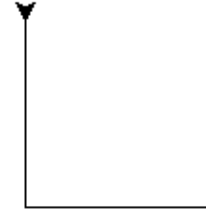
MODÜL 8

Örnek

4

right ve forward fonksiyonlarının birlikte kullanımı

```
import turtle
kalem=turtle.Turtle()
kalem.forward(-100.5)
kalem.right(90)
kalem.forward(-100.5)
turtle.done()
```



Şekil 8.5: Örnek 4 kod çıktısı

Örnek

5

right ve backward fonksiyonlarının birlikte kullanımı

```
import turtle
kalem=turtle.Turtle()
kalem.backward(-100.6)
kalem.right(90)
kalem.backward(-100.6)
turtle.done()
```



Şekil 8.6: Örnek 5 kod çıktısı

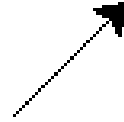
Şekil 8.3 ve Şekil 8.6 benzer çıktılar oluştuğunu fark ettiniz mi? Aynı durum Şekil 8.4 ve Şekil 8.5 için de geçerlidir. Bilgisayar: ileri 100 adım git ile geri eksi 100 adım git komutlarını aldığı anda aynı sonucu üretir. Kod çıktıları arasındaki fark mesafe girdileridir.

Örnek

6

left ve forward fonksiyonlarının birlikte kullanımı

```
import turtle
kalem=turtle.Turtle()
kalem.left(45)
kalem.forward(50)
turtle.done()
```



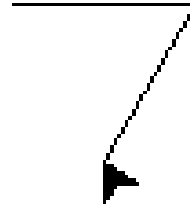
Şekil 8.7: Örnek 6 kod çıktısı

Örnek

7

Açı kullanarak şekil çizme örneği

```
import turtle
kalem=turtle.Turtle()
kalem.forward(50)
kalem.right(120)
kalem.forward(50)
kalem.right(120)
turtle.done()
```



Şekil 8.8: Örnek 7 kod çıktısı

Temel hareketler için bu örnekleri uyguladıktan sonra hareket fonksiyonlarının açı ve uzunlukları kullanımdan alacağınız örnekleri inceleyebilirsiniz.

HATIRLATMA

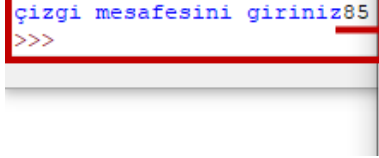
Kullanıcıdan **input()** fonksiyonu ile alınan metin değerini int veya float değerine dönüştürülmesi gerekir aksi durumda hata mesajı karşılaşılr.

MODÜL 8

Örnek 8

Çizgi mesafesini kullanıcıdan alıp şekil çizen program

```
import turtle
kalem=turtle.Turtle()
mesafe=float(input("çizgi mesafesini giriniz"))
kalem.forward(mesafe)
kalem.right(90)
kalem.forward(mesafe)
turtle.done()
```



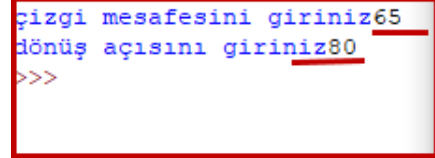
```
çizgi mesafesini giriniz85
>>>
```

Şekil 8.9: Örnek 8 kod çıktısı

Örnek 9

Çizgi mesafesini ve dönüş açısını kullanıcıdan alıp şekil çizen program

```
import turtle
kalem=turtle.Turtle()
mesafe=int(input("çizgi mesafesini giriniz"))
donus_ açısı=int(input("dönüş açısını giriniz"))
kalem.forward(mesafe)
kalem.right(donus_ açısı)
kalem.forward(mesafe)
turtle.done()
```



```
çizgi mesafesini giriniz65
dönüş açısını giriniz80
>>>
```

Şekil 8.10: Örnek 9 kod çıktısı

8.3. İşaretçi ve Çizim Araçları

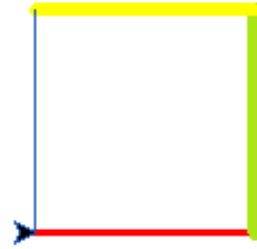
Turtle nesnesinde çizim için kalem rengi: **pencolor(" ")** veya **color(" ")** girdi olarak iki tırnak içinde renk adı "red" veya tırnak içinde diyez ile birlikte html renk kodu "#ACE515" gibi alır. Renk kodlarını <https://htmlcolorcodes.com/> adresinden öğrenebilirsiniz. Kalem kalınlığı **pensize (sayı değeri)** fonksiyonu ile ayarlanır.

Örnek

10

pencolor, pensize ve color uygulaması

```
import turtle
kalem=turtle.Turtle()
kalem.pencolor("red")
kalem.pensize(3)
kalem.forward(100)
kalem.left(90)
kalem.color("#ACE515")
kalem.pensize(8)
kalem.forward(100)
kalem.left(90)
kalem.pencolor("yellow")
kalem.pensize(6)
kalem.forward(100)
kalem.left(90)
kalem.color("#2259C1")
kalem.pensize(1)
kalem.forward(100)
kalem.left(90)
turtle.done()
```



Şekil 8.11: Örnek 10 kod çıktısı

MODÜL 8

Kaplumbağa grafiklerde pencerenin istenen noktasına gidebilme **goto (x eksen, y eksen)**

Kaplumbağa grafiklerde nokta çizme **dot()**

Kaplumbağa grafiklerde çizmeden ilerleme **penup()** veya **up()** ardından **forward()**

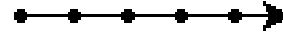
Kaplumbağa grafiklerde çizime tekrar dönmek için ilerleme **pendown()** veya **down()** ardından **forward()** fonksiyonları kullanılır.

Turtle çizim aracının görünüm şekli için de **shape()** fonksiyonu bulunmaktadır. Bu fonksiyon **'arrow'**, **'classic'**, **'turtle'**, **'circle'** olmak üzere 4 şekle sahip olabilir.

Örnek 11

dot ile çizim uygulaması

```
import turtle
kalem=turtle.Turtle()
for i in range(5):
    kalem.dot()
    kalem.forward(20)
turtle.done()
```



Şekil 8.12: Örnek 11 kod çıktısı

Örnek 12

dot ile çizgisiz şekil uygulaması

```
import turtle
kalem=turtle.Turtle()
kalem.up()
for i in range(5):
    kalem.dot()
    kalem.forward(20)
turtle.done()
```



Şekil 8.13: Örnek 12 kod çıktısı

Örnek

13

shape ve goto fonksiyonlarının kullanımı

```
import turtle
kalem=turtle.Turtle()
kalem.shape("turtle")
kalem.forward(100)
kalem.penup()
kalem.goto(0,100)
for i in range(5):
    kalem.dot()
    kalem.forward(20)
turtle.done()
```



Şekil 8.14: Örnek 13 kod çıktısı

Örnek

14

shape ve goto fonksiyonlarının farklı bir çizim için kullanımı

```
import turtle
kalem=turtle.Turtle()
kalem.shape("turtle")
for i in range(4):
    kalem.up()
    kalem.forward(20)
    kalem.dot()
    kalem.down()
    kalem.forward(20)
    kalem.dot()
turtle.done()
```



Şekil 8.15: Örnek 14 kod çıktısı

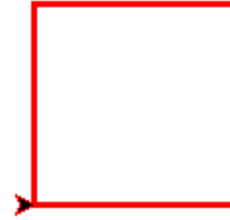
8.4. Turtle ile Geometrik Şekiller Çizme

Turtle nesnesi ile geometrik şekiller çizmek oldukça kolaydır. Çokgenler başta olmak üzere birçok geometrik şekli ileri ve sağ/sol dönüş fonksiyonları ile yapmak mümkündür. Örnek 15'te bir kare çizdirildiğini fark etmişsinizdir. Fark etmenizi istediğimiz bir diğer nokta, forward ve left fonksiyonlarının 4 defa kullanılmasıdır (Örnek 15). Kodu defalarca yazmak yerine döngüler konusunda işlendiği gibi for yapısı ile kolayca çizdirebiliriz. Aynı mantığı Örnek 16'da gösterildiği gibi altıgen çizdirmek için kullanabiliriz.

Örnek 15

turtle ile kare çizme uygulaması

```
import turtle
kalem=turtle.Turtle()
kalem.pencolor("red")
kalem.pensize(3)
for i in range(4):
    kalem.forward(100)
    kalem.left(90)
turtle.done()
```

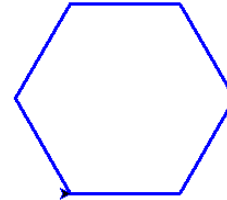


Şekil 8.16: Örnek 15 kod çıktısı

Örnek 16

turtle ile altıgen çizme uygulaması

```
import turtle
kalem=turtle.Turtle()
kalem.pencolor("blue")
kalem.pensize(3)
for i in range(6):
    kalem.forward(100)
    kalem.left(60)
turtle.done()
```



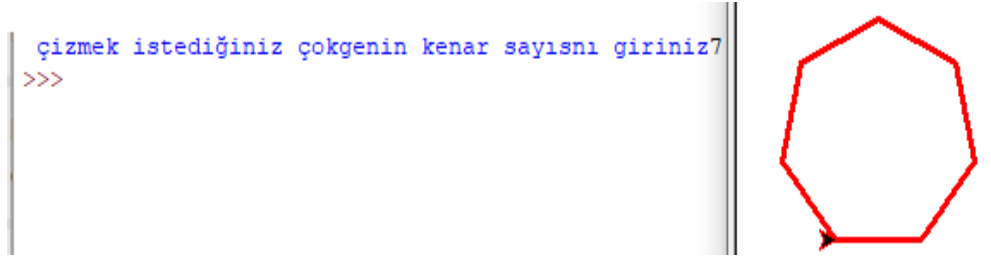
Şekil 8.17: Örnek 16 kod çıktısı

Çokgen çizdirmeyi kenar sayısını kullanıcıdan isteyerek de yapabilirsiniz. Burada dikkat edilecek nokta çokgen kaç kenarlı olursa olsun çokgen çizimi tamamlandığında 360 derecelik dönüş yapacak olmasıdır. Bu nedenle çokgenin dönüş açısı **360/ kenar sayısı** olur.

Örnek**17****Kullanıcının istediği kenar sayısında çokgen çizdirmek**

```
import turtle
kalem=turtle.Turtle()
kalem.pencolor("red")
kalem.pensize(3)

kenar_sayısı=int(input(" çizmek istediğiniz çokgenin kenar sayısını giriniz"))
for i in range(kenar_sayısı):
    kalem.forward(50)
    kalem.left(360/kenar_sayısı)
turtle.done()
```



Şekil 8.18: Örnek 17 kod çıktısı

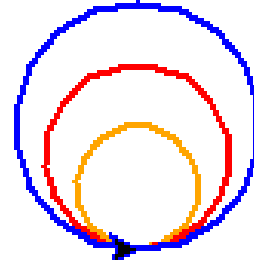
Çember çizmek için **circle (yarıçap değeri)** fonksiyonu yeterlidir. Örnek 18’de farklı renkte ve yarıçapta 3 adet çember çizdirilmektedir.

MODÜL 8

Örnek 18

circle fonksiyonu ile 3 farklı yarıçapta ve renkte çember çizdirme

```
import turtle
kalem=turtle.Turtle()
kalem.pencolor("orange")
kalem.pensize(2)
kalem.circle(20)
kalem.pencolor("red")
kalem.circle(30)
kalem.pencolor("blue")
kalem.circle(40)
turtle.done()
```



Şekil 8.19: Örnek 18 kod çıktısı

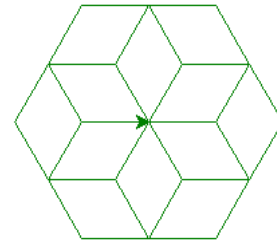
Çember ve çokgenler kullanılarak çok farklı desenler oluşturabileceğini fark ettiniz mi?

Çokgenlerle farklı desenler oluşturmaya dair Örnek 19'u inceleyebilirsiniz. Bunu yapmak için iç içe döngüler kullanılmaktadır. Yani çokgenlerden bol miktarda çizdirilmesi gerekmektedir. Bu kodlar robota yaptırıldığında istenilen desenler farklı yüzeylere çizdirilebilirdi.

Örnek 19

İç içe döngüler ile desen çizdirme

```
import turtle
kalem=turtle.Turtle()
kalem.color("green")
for i in range (6):
    for j in range (6):
        kalem.forward(50)
        kalem.left(60) # iç döngü
    kalem.left(60) # dış döngü
turtle.done()
```



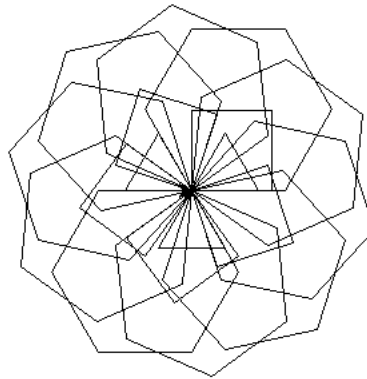
Şekil 8.20: Örnek 19 kod çıktısı

Örnek 19'daki deseni istendiğinde ve parametreler girildiğinde çizen bir fonksiyon tanımlayıp kullanınız.

Örnekte fonksiyon için verilen değerlerin doğruluğunun kontrol edildiğine dikkat ediniz. Kenar uzunluğunun eksi olarak da girilebileceği düşünüldüğünden ve if koşul yapısının karmaşık hâle getirilmemesi için dâhil edilmemiştir.

Örnek**20****Desen çizen fonksiyon uygulaması**

```
import turtle
def desen_çiz (kenar_uzunluğu=50,iç_kenar=3,tur_sayısı=3):
    if(tur_sayısı <01 or iç_kenar<3):
        print("hatalı veri girdiniz")
    else:
        kalem=turtle.Turtle()
        for i in range (tur_sayısı):
            for j in range (iç_kenar):
                kalem.forward(kenar_uzunluğu)
                kalem.left(360/iç_kenar)
            kalem.left(360/tur_sayısı)
desen_çiz()
desen_çiz(60,4,5)
desen_çiz(70,6,10)
turtle.done()
```



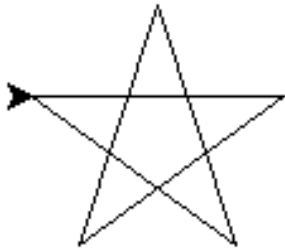
Şekil 8.21: Örnek 20 kod çıktısı

MODÜL 8

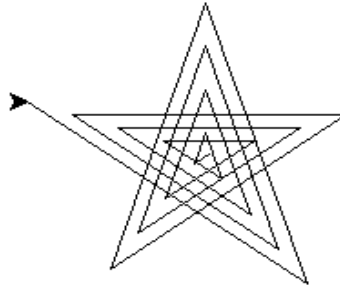
8.5. Bölüm Sonu Örnekleri

Resimlerde gösterilen şekilleri çizecek Python kodlarını yazınız.

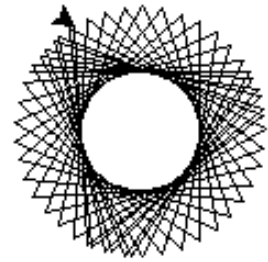
Soru-1



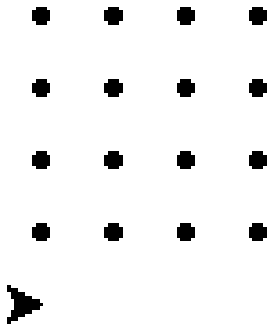
Soru-2



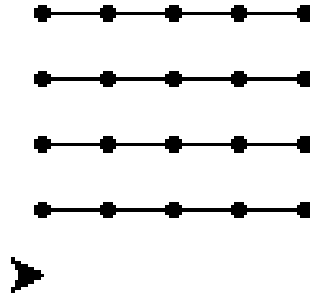
Soru-3



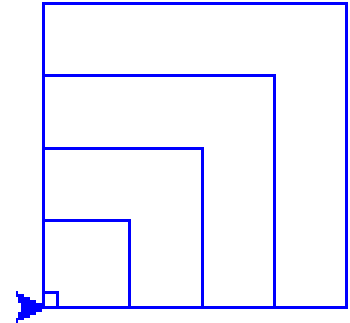
Soru-4



Soru-5



Soru-6



Cevaplar

1.

```
import turtle
yıldız = turtle.Turtle()
for i in range(5):
    yıldız.forward(100)
    yıldız.right(144)
turtle.done()
```

2.

```
import turtle
yıldız = turtle.Turtle()
for i in range(20):
    yıldız.forward(i * 10)
    yıldız.right(144)
turtle.done()
```

3.

```
import turtle
yıldız = turtle.Turtle()
for i in range(50):
    yıldız.forward(100)
    yıldız.right(123)
turtle.done()
```

4.

```
import turtle
kalem = turtle.Turtle()
kalem.penup()
for y in range(4):
    for i in range(4):
        kalem.dot()
        kalem.forward(20)
    kalem.backward(80)
    kalem.right(90)
    kalem.forward(20)
    kalem.left(90)
turtle.done()
```

5.

```
import turtle
kalem = turtle.Turtle()
for y in range(4):
    kalem.down()
    kalem.dot()
    for i in range(4):
        kalem.forward(20)
        kalem.dot()
    kalem.up()
    kalem.backward(80)
    kalem.right(90)
    kalem.forward(20)
    kalem.left(90)
turtle.done()
```

MODÜL 8

6.

```
import turtle
kalem = turtle.Turtle()
kalem.color("blue")
for k in range(5,106,25):
    print(k)
    for i in range(4):
        kalem.forward(k)
        kalem.left(90)
turtle.done()
```